

# ***Innocent Code***

*A Security Wake-Up Call for Web Programmers*

Sverre H. Huseby



John Wiley & Sons, Ltd

*This page intentionally left blank*

“This book is much more than a wake-up call. It is also an eye-opener. Even for those who are already awake to the problems of Web server security, it is a serious guide for what to do and what not to do.”

*Peter G. Neumann, risks.org*

*This page intentionally left blank*

# ***Innocent Code***

*A Security Wake-Up Call for Web Programmers*

Sverre H. Huseby



John Wiley & Sons, Ltd

Copyright © 2004

John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester,  
West Sussex PO19 8SQ, England

Telephone (+44) 1243 779777

Email (for orders and customer service enquiries): [cs-books@wiley.co.uk](mailto:cs-books@wiley.co.uk)

Visit our Home Page on [www.wileyeurope.com](http://www.wileyeurope.com) or [www.wiley.com](http://www.wiley.com)

All Rights Reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except under the terms of the Copyright, Designs and Patents Act 1988 or under the terms of a licence issued by the Copyright Licensing Agency Ltd, 90 Tottenham Court Road, London W1T 4LP, UK, without the permission in writing of the Publisher, with the exception of any material supplied specifically for the purpose of being entered and executed on a computer system for exclusive use by the purchase of the publication. Requests to the Publisher should be addressed to the Permissions Department, John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, England, or emailed to [permreq@wiley.co.uk](mailto:permreq@wiley.co.uk), or faxed to (+44) 1243 770620.

This publication is designed to provide accurate and authoritative information in regard to the subject matter covered. It is sold on the understanding that the Publisher is not engaged in rendering professional services. If professional advice or other expert assistance is required, the services of a competent professional should be sought.

### *Other Wiley Editorial Offices*

John Wiley & Sons Inc., 111 River Street, Hoboken, NJ 07030, USA

Jossey-Bass, 989 Market Street, San Francisco, CA 94103-1741, USA

Wiley-VCH Verlag GmbH, Boschstr. 12, D-69469 Weinheim, Germany

John Wiley & Sons Australia Ltd, 33 Park Road, Milton, Queensland 4064, Australia

John Wiley & Sons (Asia) Pte Ltd, 2 Clementi Loop #02-01, Jin Xing Distripark, Singapore 129809

John Wiley & Sons Canada Ltd, 22 Worcester Road, Etobicoke, Ontario, Canada M9W 1L1

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic books.

### *Library of Congress Cataloging-in-Publication Data*

Huseby, Sverre H.

Innocent code : a security wake-up call for Web programmers / Sverre  
H. Huseby.

p. cm.

"A Wiley-Interscience publication."

ISBN 0-470-85744-7

1. Computer security. 2. Computer networks--Security measures. 3.

World Wide Web--Security measures. I. Title.

QA76.9.A25H88 2003

005.8--dc22

2003015774

### *British Library Cataloguing in Publication Data*

A catalogue record for this book is available from the British Library

ISBN 0-470-85744-7

Typeset in 10.5/13pt Sabon by Laserwords Private Limited, Chennai, India

Printed and bound in Great Britain by Biddles Ltd, Guildford and King's Lynn

This book is printed on acid-free paper responsibly manufactured from sustainable forestry in which at least two trees are planted for each one used for paper production.

# Contents

Foreword	ix
Acknowledgments	xi
Introduction	xiii
0.1 The Rules	xiv
0.2 The Examples	xv
0.3 The Chapters	xvi
0.4 What is Not in This Book?	xvii
0.5 A Note from the Author	xviii
0.6 Feedback	xviii
<b>1 The Basics</b>	<b>1</b>
1.1 HTTP	1
1.1.1 Requests and responses	2
1.1.2 The Referer header	6
1.1.3 Caching	7
1.1.4 Cookies	9
1.2 Sessions	10
1.2.1 Session hijacking	11
1.3 HTTPS	15
1.4 Summary	19
1.5 Do You Want to Know More?	19
<b>2 Passing Data to Subsystems</b>	<b>21</b>
2.1 SQL Injection	22
2.1.1 Examples, examples and then some	22
2.1.2 Using error messages to fetch information	30

2.1.3	Avoiding SQL injection	33
2.2	Shell Command Injection	39
2.2.1	Examples	40
2.2.2	Avoiding shell command injection	42
2.3	Talking to Programs Written in C/C++	48
2.3.1	Example	48
2.4	The Evil Eval	50
2.5	Solving Metacharacter Problems	50
2.5.1	Multi-level interpretation	52
2.5.2	Architecture	53
2.5.3	Defense in depth	54
2.6	Summary	55
<b>3</b>	<b>User Input</b>	<b>57</b>
3.1	What is Input Anyway?	57
3.1.1	The invisible security barrier	62
3.1.2	Language peculiarities: totally unexpected input	65
3.2	Validating Input	67
3.2.1	Whitelisting vs. blacklisting	71
3.3	Handling Invalid Input	74
3.3.1	Logging	76
3.4	The Dangers of Client-side Validation	79
3.5	Authorization Problems	82
3.5.1	Indirect access to data	83
3.5.2	Passing too much to the client	86
3.5.3	Missing authorization tests	90
3.5.4	Authorization by obscurity	91
3.6	Protecting server-generated input	92
3.7	Summary	95
<b>4</b>	<b>Output Handling: The Cross-site Scripting Problem</b>	<b>97</b>
4.1	Examples	98
4.1.1	Session hijacking	99
4.1.2	Text modification	103
4.1.3	Socially engineered Cross-site Scripting	104
4.1.4	Theft of passwords	108
4.1.5	Too short for scripts?	109
4.2	The Problem	111
4.3	The Solution	112
4.3.1	HTML encoding	113
4.3.2	Selective tag filtering	114
4.3.3	Program design	120
4.4	Browser Character Sets	121
4.5	Summary	122
4.6	Do You Want to Know More?	123
<b>5</b>	<b>Web Trojans</b>	<b>125</b>
5.1	Examples	125
5.2	The Problem	130



5.3 A Solution	131
5.4 Summary	133
<b>6 Passwords and Other Secrets</b>	<b>135</b>
6.1 Crypto-Stuff	135
6.1.1 Symmetric encryption	137
6.1.2 Asymmetric encryption	137
6.1.3 Message digests	139
6.1.4 Digital signatures	140
6.1.5 Public key certificates	141
6.2 Password-based Authentication	142
6.2.1 On clear-text passwords	142
6.2.2 Lost passwords	144
6.2.3 Cracking hashed passwords	146
6.2.4 Remember me?	150
6.3 Secret Identifiers	151
6.4 Secret Leakage	153
6.4.1 GET request leakage	154
6.4.2 Missing encryption	156
6.5 Availability of Server-side Code	157
6.5.1 Insecure file names	157
6.5.2 System software bugs	158
6.6 Summary	160
6.7 Do You Want to Know More?	161
<b>7 Enemies of Secure Code</b>	<b>163</b>
7.1 Ignorance	163
7.2 Mess	165
7.3 Deadlines	171
7.4 Salesmen	173
7.5 Closing Remarks	174
7.6 Do You Want to Know More?	174
<b>8 Summary of Rules for Secure Coding</b>	<b>177</b>
<b>Appendix A Bugs in the Web Server</b>	<b>187</b>
<b>Appendix B Packet Sniffing</b>	<b>193</b>
B.1 Teach Yourself TCP/IP in Four Minutes	193
B.2 Sniffing the Packets	195
B.3 Man-In-The-Middle Attacks	196
B.4 MITM with HTTPS	197
B.5 Summary	198
B.6 Do You Want to Know More?	198
<b>Appendix C Sending HTML Formatted E-mails with a Forged Sender Address</b>	<b>199</b>

Appendix D	More Information	201
	D.1 Mailing Lists	201
	D.2 OWASP	203
Acronyms		205
References		209
Index		221

# Foreword

There has been a rude awakening for the IT industry in the last few years. For nearly a decade corporations have been told by the media and consultants that they needed firewalls, intrusion detection systems and network scanning tools to stop the barrage of cyber attacks that we all read about daily. Hackers are stealing credit cards, booking flights to exotic locations for free and downloading personal information about the latest politicians' affair with an actress. We have all seen the stories and those of us with an inquisitive mind have all wondered how it really happens.

As the information security market grew into a vast commercial machine pushing network and operating system security technology and processes as the silver bullet to cure all ills, the IT industry itself grew in a new direction. Business leaders and marketing managers discovered that the lowest common denominator to any user (or potential user) is the web browser, and quite frankly why in the world wouldn't they want to appeal to all the possible clients out there? Why would you want to restrict the possibility of someone signing up for your service? Web enabling applications and company data was not just a trend, it has been a phenomena. Today there are web interfaces to almost all major applications from development source code systems to human resources payroll systems and sales tracking databases. When we browse the Web and the local weather is displayed so conveniently in the side-menu, it's a web application that put it there. When we check our online bank balance, it's a system of complex web applications that compute and display the balance.

Creating these vast complex pieces of technology is no trivial task. From a technology stance, Microsoft and Sun are leading the charge with platforms

and supporting languages that provide flexible and extensible bases from which to build. With flexibility comes choice, and whilst it is true that these platforms can provide excellent security functionality, the security level is a choice of the designer and developer. All of the platforms on offer today can equally create secure and insecure applications, and as with many things in life, the devil is in the details. When building a web application the details are almost exclusively the responsibility of the developer.

This book takes a unique and highly effective approach to educating the people that can effect a change by addressing the people who are actually responsible for writing code; the developers themselves. It is written by a developer for developers, which means it speaks the developer lingo and explains issues in a way that as a developer you will understand. By taking a pragmatic approach to the issue, the author walks you, the reader, through an overview of the issues and then delves into the devilish details supporting issues with examples and real life scenarios that are both easy to understand and easy to realize in your own code.

This book is a serious must have for all developers who are building web sites. I know you will enjoy it as much as I did.

Mark Curphey

*Mark Curphey has a Masters degree in Information Security and runs the Open Web Application Security Project. He moderates the sister security mailing list to Bugtraq called webappsec that specializes in web application security. He is a former Director of Information Security for Charles Schwab, consulting manager for Internet security Systems and veteran of more banks and consulting clients than he cares to remember.*

# Acknowledgments

This book would have been less readable, less consistent, and more filled with bugs if it wasn't for a handful of smart friends and colleagues that helped me pinpoint troublesome areas along the way. All I did was to promise them a beer and honorable mention in this section, and they started spending hours and days (and some even weeks) helping me out.

First of all, Jan Ingvoldstad has spent an amazing amount of time reading, commenting, and suggesting improvements to almost every paragraph.

In addition, the following people have spent quite some time reading and commenting on early versions of the text: Lars Preben S. Arnesen, Erik Assum, Jon S. Bratseth, Per Otto Christensen, Per Kristian Gjermshus, Morten Grimnes, Leif John Korshavn, Rune Offerdal, Frode Sandnes, Frank Solem, Rune Steinberg, Kent Vilhelmsen and Sigmund Øy.

Kjetil Valstadsve made me rethink some sections, and Tore Anderson, Kjetil Barvik, Maja Bratseth, Lasse G. Dahl, Dennis Groves, Jan Kvile, Filip van Laenen, Glenn T. Lines, Kevin Spett, Thorkild Stray and Bjørn Stærk gave valuable feedback and ideas to parts of the text.

Please note that none of the people on this list of gratitude should be blamed for any errors or omissions whatsoever in this book. I was stupid enough not to follow all the advice given to me by these kind and experienced people, so I'm the only one to blame if you feel like blaming anyone for anything (concerning this book, that is).

I would also like to thank my editor Gaynor Redvers-Mutton and her friends at Wiley for believing in my book proposal even though most of their reviewers wanted to turn the book into a traditional infrastructure security thing. : - )

As I find book dedications quite meaningless, I'd rather say "hi" to Markus and Matilde in this section. Thanks for giving me good memories while you keep me busy throughout the days.

And last, but certainly not least, I bow deeply for my beloved wife, Hanne S. Finstad. She always makes me feel safe and free of worries. Without that kind of support (which I'm not sure she knows she's giving me), I would never have been able to write a book (cliche, but true anyway). She's the most creative, intelligent, beautiful, . . . oh, sorry. I'll tell her face to face instead.

S. H. H.

# Introduction

This book is kind of weird. It's about the security of a web site, but it hardly mentions firewalls. It's about the security of information, but it says very little about encryption. So what's this book all about? It describes a small, and often neglected, piece of the web site security picture: Program code security.

Many people think that a good firewall, encrypted communication and staying up to date on software patches is all that is needed to make a web site secure. They're wrong. Many of today's web sites contain program code that make them dynamic. Code written using tools such as Java, PHP, Perl, ASP/VBScript, Zope, ColdFusion, and many more. Far too often, this code is written by programmers who seem to think that security is handled by the administrators. The effect is that an enormous number of dynamic web sites have logical holes in them that make them vulnerable to all kinds of nasty attacks. Even with both firewall and encryption in place.

Current programmer education tends to see security as off topic. Something for the administrators, or for some elite of security specialists. We learn how to program. Period. More specifically, to make programs that please the customers by offering the requested functionality. Some years ago, that would probably suffice. Back then, programs were internal to organizations. Every person with access to our program wanted it to operate correctly, so that they could do their day to day job.

In the age of the Web, however, most of us get to create programs that are available to the entire world. Legitimate users still just want the program to do its job for them. Unfortunately, our program is also available to lots of people who find amusement in making programs break. Or better, making them do things they were not supposed to do.